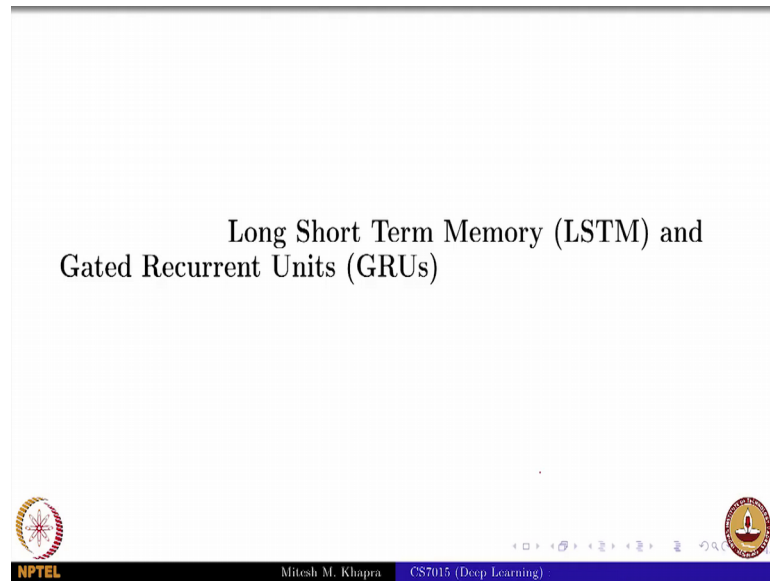


Deep Learning
Prof. Mitesh M. Khapra
Department of Computer Science Engineering
Indian Institute of Technology, Madras

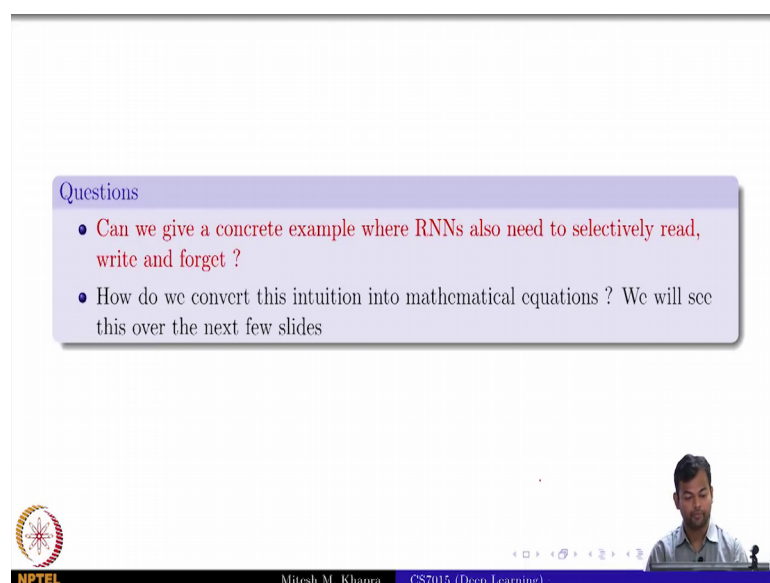
Lecture – 109
Long Short Term Memory (LSTM) and Gated Recurrent Units (GRUs)

(Refer Slide Time: 00:11)



So, with that motivation let us go to the next module, where we will talk about Long Short Term Memory and Gated Recurrent Units ok

(Refer Slide Time: 00:21)



So now, all this was fine in terms of ok. I gave you a derivation on the board and say that, this is not required, but can I give you a concrete example, where RNNs also need to selectively read, write and forget right only then you will be convinced that this kind of morphing is bad in the case of RNNs. So, I will start with that example and then once we agree that, we need selective read, write and forget. How do we convert this into some mathematical equations? Right, because conceptually it is fine, but you have to write some equations. So that the RNN can do some computations, where you have selective read write and forget right. So, that is what we are going to do over the rest of the lecture.

(Refer Slide Time: 00:55)

Review: The first half of the movie was dry but the second half really picked up pace. The lead actor delivered an amazing performance

- Consider the task of predicting the sentiment (positive/negative) of a review
- RNN reads the document from left to right and after every word updates the state
- By the time we reach the end of the document the information obtained from the first few words is completely lost
- Ideally we want to
 - **forget** the information added by stop words (a, the, etc.)
 - **selectively read** the information added by previous sentiment bearing words (awesome, amazing, etc.)
 - **selectively write** new information from the current word to the state

Mitesh M. Khapra CS7015 (Deep Learning) 13/43

So, first let me start with the concrete example, where you want to predict the sentiment of a review using an RNN. So, this is the RNN structure, we have done this in the past that you have a sentence, 1 word at a time is your every time step, you will feed this to the RNN and at the last time step, you will make a prediction. And as I said the RNN needs a document from left to right and by the time it reaches the end, the information obtained from the first few words is completely lost right because, it is a long document and you are continuously writing to the same cell state.

So, you will lose the information that, you had gained at the previous time step. But, ideally we want to do the following, we want to forget the information added by stop

words like a, an, the these do not contribute to the sentiment of the I can ignore these words and still figure out the sentiment of the document.

I want to selectively read the information added by previous sentiment bearing words. So, when I have reach the last time step, I should be able to read everything else, which had some sentiments before it and focus on those words just I want to selectively read from these sentiment bearing words and also I want to selectively write the new information. So, I have read the word performance, now I want to selectively write it to the memory, whether I should write it completely or should I only write parts of it or not that is what I need to decide. So, that is fair, this is a typical example, where RNN also when it is dealing with long documents, it needs to understand what is the important information in the document that needs to be retained and then selectively read, write and forget ok.

So, I am spending a lot of time on this analogy, because you need to really understand that this is important and this is where RNN suffer right, if you are using them for very very long documents, if we have document of the size 1000 words, which is not which is not uncommon right because Wikipedia pages have much more than that per document. So, it is going to be very hard to encode the entire document using an RNN not that it is going to become significantly easier with LSTM or GRUS, but to certain extent it will become easier ok

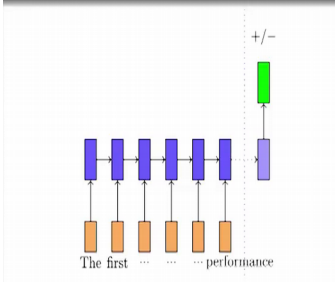
(Refer Slide Time: 02:47)

Questions

- Can we give a concrete example where RNNs also need to selectively read, write and forget ?
- How do we convert this intuition into mathematical equations ?

Mitesh M. Khapra CS7015 (Deep Learning)

(Refer Slide Time: 02:53)



The first ... performance

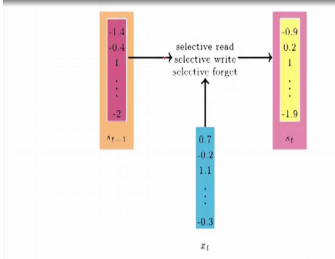
Review: The first half of the movie was dry but the second half really picked up pace. The lead actor delivered an amazing performance

- Recall that the blue colored vector (s_t) is called the state of the RNN
- It has a finite size ($s_t \in \mathbb{R}^n$) and is used to store all the information upto timestep t
- This state is analogous to the whiteboard and sooner or later it will get overloaded and the information from the initial states will get morphed beyond recognition
- **Wishlist:** selective write, selective read and selective forget to ensure that this finite sized state vector is used effectively

Mitesh M. Khapra CS7015 (Deep Learning) 15/43

Now the next part is how do we convert this intuition into some mathematical equations? Right so, let us look at that. So, in this diagram recall that the blue colored vector is called the state of the RNN, it has a finite size, so now I will just call it as s_t belongs to some \mathbb{R}^n and the state is analogous to the whiteboard and sooner or later it will get overloaded with information and we need to take care of this right. So now, our wish list is selectively read write and forget ok. So, let us start with that.

(Refer Slide Time: 03:15)



s_{t-1} x_t s_t

selective read
selective write
selective forget

- Just to be clear, we have computed a state s_{t-1} at timestep $t - 1$ and now we want to overload it with new information (x_t) and compute a new state (s_t)
- While doing so we want to make sure that we use selective write, selective read and selective forget so that only important information is retained in s_t
- We will now see how to implement these items from our wishlist

Mitesh M. Khapra CS7015 (Deep Learning) 16/43

So, what we want to do is that and this is the problem definition, now that we have computed the state of the RNN, this is a blue colored vector although, it is not blue, but this the blue colored vector from the previous diagram, where the state of the RNN was computed. I know what the state is at time step $s_t - 1$, now I want from here to here go from here to here; that means, from $s_t - 1$. I want to compute the new state of the RNN right. So, I had something written on the whiteboard, I want to write something new, I want to change the state of the whiteboard and this is the new information that has coming to me right the x_t is the new information at time step t .

And while doing this I want to make sure that I use selectively write, read and forget. So, these 3 operations have to come in somewhere in the between. So, that I am true or faithful to the analogy, which I have been giving right that is the this is the our problem definition, now going from $s_t - 1$ to s_t and introducing these 3 operations along the way that is what we are interested in doing.

(Refer Slide Time: 04:21)

Selective Write

- Recall that in RNNs we use s_{t-1} to compute s_t
 $s_t = \sigma(Ws_{t-1} + Ux_t)$ (ignoring bias)
- But now instead of passing s_{t-1} as it is to s_t we want to pass (write) only some portions of it to the next state
- In the strictest case our decisions could be binary (for example, retain 1st and 3rd entries and delete the rest of the entries)

17/43

I will go one by one; we will implement each of these 3 items right. So, we will start with selective write. So, recall that in RNNs this is what happens at every time step t , you take the previous time, step previous cell state, you take the current input do you recognize the operation here? How many of you recognize the operation here? Raise your hands ok. So, this is nothing, but the following operation and as usual I have ignored the bias is that fine. So, that is what I am representing it as.

But now so, one way of looking at it is that, when I am computing s_t , I am reading or I am taking the whole of s_{t-1} . So, once I have computed s_{t-1} , I am writing this to my whiteboard and then whole of it would be used to compute the next time step ok, but instead of doing this what I want is, I want to read only selective portions of s_{t-1} or rather I want to write only selective portions of s_{t-1} . Once I have computed s_{t-1} , I do not want to write the whole of it, because then the whole of it will be used to compute the next cell state, I do not want that I just want to selectively write some portions of it ok.

Now, in the strictest case since, I know that s_{t-1} belongs to \mathbb{R}^n . It is an n dimensional vector in the strictest case, what I could have done is I could have used a binary decision that of all these n entries, I am going to read some entries and ignore the others. So, all the other entries I am going to set to 0 fine, that is the strictest thing, that you could have done. Now for any of these strictest things, what is the soft solution? So, for binary what is the soft solution? Binary 0 to 1. So, what is the soft solution for that between?

Student: (Refer Time: 06:05)

0 to 1 so, read some fraction of each of these dimensions right. So, let us try to understand what I am trying to do here ok. So, and the third bullet some of these entries should have gone to 0 right ok.

So, instead of doing this, what we want to do is we have this vector, which has n entries this is the cell state at $t-1$. I do not want to write the entire vector onto the final cell state, what I want to do is, I will take some fractions of it is say 0.2 of this 0.3 of this 0.4 of these and then write only that, do you see the operation that I am trying to do right. I want to take some fractions and write only those to the cell and as I said this is the softer version of the hard decision, which would have been 0 for this 1, for this again 0 for this and so on right.

(Refer Slide Time: 07:07)

Selective Write

- Recall that in RNNs we use s_{t-1} to compute s_t
 $s_t = \sigma(Ws_{t-1} + Ux_t)$ (ignoring bias)
- But now instead of passing s_{t-1} as it is to s_t we want to pass (write) only some portions of it to the next state
- In the strictest case our decisions could be binary (for example, retain 1st and 3rd entries and delete the rest of the entries)
- But a more sensible way of doing this would be to assign a value between 0 and 1 which determines what fraction of the current state to pass on to the next state

Mitesh M. Khapra CS7015 (Deep Learning) 17/43

How to do this? Why to do this? All that is not clear, I am just telling you the intuition, how and why will become clear later is that fine So, we want to be able to take s_{t-1} and write only selective portions of it or pass only selective portions of it to s_t .

(Refer Slide Time: 07:23)

Selective Write

- We introduce a vector o_{t-1} which decides what fraction of each element of s_{t-1} should be passed to the next state
- Each element of o_{t-1} gets multiplied with the corresponding element of s_{t-1}
- Each element of o_{t-1} is restricted to be between 0 and 1
- But how do we compute o_{t-1} ? How does the RNN know what fraction of the state to pass on?

Mitesh M. Khapra CS7015 (Deep Learning) 18/43

So, whenever we compute s_t , we do not want to write the whole of s_{t-1} , just want to use selective portions of that. So, what we do is we introduce something known as a gate and. So, this gate is o_{t-1} ok, we take the original cell state s_{t-1} do an element wise product with a gate, which is known as the output gate and then write that

product to a new vector, which is $s_t - 1$ ok. So, initially this will look confusing, but it will become clear, by the end of this lecture ok. So, is that fine, this is what I am trying to do again, how to do this is not clear, but this still matches the intuition, which I have been trying to build that I want to write only selective portion of the data, which I already have is that fine ok. So, each element of o_{t-1} gets multiplied by the corresponding element of $s_t - 1$ and it decides what fraction is going to be copied. And this o_{t-1} is going to be, between 0 to 1.

But how do I compute o_{t-1} , how does the RNN know what fraction of the cell state to get to the next state? How will it do it? We need to learn something whenever you want to learn something, what do we introduce? Everyone?

Student (Refer Time: 08:26)

Parameter sorry, what did you guys say, back propagation. Back propagation will do what? It will work in the air or propagate to what?

Student: (Refer Time: 08:37).

Whenever you want to do some kind of a learning, I want to learn some function, what do I introduce?

Student: (Refer Time: 08:42) parameter.

Parameter right so, that is what we are going to do, we are now going to introduce a parametric form for o_{t-1} right. And, remember this throughout in machine learning, whenever you want to learn something always introduce a parametric form of that quantity and then learn the parameters of that function, do you get this how many of you get the statement? Ok this is what we have been saying day from right from class 2 or class 3 right.

(Refer Slide Time: 09:13)

selective write

Selective Write

- Well the RNN has to learn o_{t-1} along with the other parameters (W, U, V)
- We compute o_{t-1} and h_{t-1} as

$$o_{t-1} = \delta(W_o h_{t-2} + U_o x_{t-1} + b_o)$$

$$h_{t-1} = o_{t-1} \odot \sigma(s_{t-1})$$

o_{t-1}

$y = W^T x$
 $= \text{DNN}(x)$

$$y = \text{LSM}(x)$$

R R R F R R R
The movie was long but really amazing

19/43

Always introduce a parametric function for your input and output and learn the parameters of this function. So, that is exactly, what I am going to do, I am going to say that o_{t-1} is actually, this function I am just giving some time to digest this. So, this is at time step $t-1$. So, it depends on the input at time step $t-1$. It also depends on the output means whatever comes out of this right. So, the same operation, what have happened at time step $t-2$. So, whatever was the output at that state it will also depend on this.

You just take a while to digest, this equation you will see at least 6 more equations of this form in this lecture. So, if you are comfortable with one all of them, would be clear. So, try to connect the whole story, I have s_{t-1} , I do not want to pass it as on pass it on as it is to s_t . So, I am computing some intermediate value, where I will only selectively write some portions of s_{t-1} and selectively write in the strictest case, it should be binary, but that is not, what we are interested in. We introduce fractions if the fraction has to learn binary let it learn, but we will make it fractional; that means, we will make it between 0 to 1.

Hence, the sigmoid function right remember in one of these lectures, we had said that sigmoids are still used because in RNNs and LSTMs; remember in we had said that sigmoids are bad use (Refer Time: 10:32) or use ReLU, but we had ended with sigmoids

are still used in the case of recurrent neural networks and LSTMs. So, this is where they are used ok, how many should get that connection? Ok good fine.

So, we use sigmoids, because we want the fraction to be between 0 to 1 and we also want some parametrization right and this is the particular form that, we have chosen there are various equations possible various things, you could have done here. In fact, there are 10 to 15 different variants of LSTMs, I am covering the most popular one, which uses the following equation right. So, it is says that this is how you will compute the output gate and that gate will regulate, how much of the cell state should be passed on from t minus 1 to the next state? Ok. Everyone clear with this ok. So now, if you are clear with this give me an equation for H_{t-1} .

Student: (Refer Time: 11:22).

Loudly, everyone s_{t-1} is that fine right. So, this is the equation that we will have. So, we have done selective writing and these parameters are no special, they will be learned along with the other parameters of the network ok. So, let us spare some thought on that. You got a certain loss at the output earlier, you just had these parameters W, U , which were the parameters of RNN, which you are adjusting to learn this loss. Now in addition, you also have the flexibility to adjust these parameters.

So, that if the loss could improve by selectively writing something then, these parameters should be updated accordingly right, may be you are being over aggressive and making o_{t-1} to be all ones; that means, you are passing everything to the next state right. Now it has the chance because, they have introduce parameters, if it helps the overall loss, it better make these fractions more appropriate. So, that only selective information is passed to the next state, how many you get this intuition? So, that is why anytime, you introduce parameters, you have more flexibility in learning, whatever you intend to learn.

There is remember, one clear difference here right and that is where I said that, while I was giving the analogy, I was really setting up things, but here there is one distinction, what is the distinction? That is there ideally what would, I have wanted. Suppose I take the example of the review and the review was say, the movie was long, but really amazing ok. Now which is the word here, which is actually trying to mislead? So, overall sentiment is positive right, everyone agrees with that, but which is the word which is misleading?

Student: Long.

Long right; that means, I need to do what with that word?

Student: (Refer Time: 13:11).

Forget that would right now ideally, I would have wanted someone telling me retain, retain, retain, forget, retain, retain, retain I would have a label for each of these words and then I could have a loss function which tells me, whether my gates were actually are there in to this decisions or not? So, remember my gates are learning some distribution of t minus 1, which tells me what fraction to retain? And at this particular time step, I would have wanted t minus 1 to be all 0's ok. I would have wanted to forget, but this kind of not just t minus 1, this will become more clear, when I do all the other gates also. So, what I am trying to say, is that you should have had some supervision, which tells you which information to retain and which information to forget, but you do not have this supervision right no one is telling, whether these are the important words these are not the important words?

So, that is the difference between the whiteboard analogy, there you knew exactly which step is important and which step is not important here, you do not know that all you know is, that you have a final loss function, which depends on plus or minus, whether the this prediction is close to positive or close to negative and what is the loss and that loss is what is being back propagated, but the difference, now is that you have introduced a model, which can learn to forget some things right. Earlier you did not have a model, which could learn to write or read or forget selectively, now you have introduced a model, this is a better modelling choice right. So, the same as we have had arguments that you could do y is equal to W transpose x or you could do y is equal to deep neural network of x right, you are making different modelling choices here and with the hope that one modelling choice is better than the other choice.

So, just as RNN was one modelling choice now you are using a different modelling choice where again with the help of these gates and all you can definitely write a function of y is a function of the input and that function is going to be LSTM function, which we will see in detail. So this, one part of that function and while doing this you are just making a better modelling choice, which allows you to learn more parameters and along the way, if important do selective write, read and forget is that clear? Right so, you

would see the difference, what would have been the ideal case and what is it that you have the ideal case? Would have been explicit supervision for what to forget, read and write, you will never have that, but you are still making a modelling choice, which allow you to do that. So, if it required to model while back propagation should be able to learn these parameters. So, get you are able to do that.

(Refer Slide Time: 15:39)

Selective Write

- Well the RNN has to learn o_{t-1} along with the other parameters (W, U, V)
- We compute o_{t-1} and h_{t-1} as

$$o_{t-1} = \sigma(W_o h_{t-2} + U_o x_{t-1} + b_o)$$

$$h_{t-1} = o_{t-1} \odot \sigma(s_{t-1})$$
- The parameters W_o, U_o, b_o need to be learned along with the existing parameters W, U, V
- The sigmoid (logistic) function ensures that the values are between 0 and 1
- o_t is called the output gate as it decides how much to pass (write) to the next time step

19/43

I know I am repeating myself, but it is very important that you understand this situation, how many of you get this now? And as I said these parameters will be learned along with other parameters and o_t is called the output gate, because it decides what to output to the next cell, state still you see that there is a lot of gap here, we have not reached s_t yet we are still at s_t minus 1, we have computed some intermediate value. But we have not reached s_t yet and along the way we had 3 things selective write, read and forget, we have only taking care of selective write so, far ok.

(Refer Slide Time: 16:05)

Selective Read

- We will now use h_{t-1} to compute the new state at the next time step
- We will also use x_t which is the new input at time step t

$$\tilde{s}_t = \sigma(W h_{t-1} + U x_t + b)$$

Mitesh M. Khapra CS7015 (Deep Learning)

Now let us look at selective read. So, what this selective read, do you are going to get new information at time step t , which is x_t right and now instead of this original cell state, you have used the selectively written cell state because that is what you have written now. So, that is what you should use.

Now, using a combination of these 2, I am going to compute, some intermediate value and just stare at this equation, this equation form is very similar to the RNN equation form right only thing is that instead of s_{t-1} , I am using h_{t-1} and for good reason because, I know that h_{t-1} contains only selectively written values from h_{t-1} is that fine and x_t is the new input. Still there is some gap here, I have not reached s_t yet I am still at an intermediate value. So, this is the new input, which I have received now what should I do with this new input? Selectively read this input I do not want to take all of this input because, may be the input which I have got now is a stop word and I do not want to read all of it right, do you get that?

(Refer Slide Time: 17:15)

Selective Read

- We will now use h_{t-1} to compute the new state at the next time step
- We will also use x_t which is the new input at time step t

$$\tilde{s}_t = \sigma(W h_{t-1} + U x_t + b)$$

- Note that W, U and b are similar to the parameters that we used in RNN (for simplicity we have not shown the bias b in the figure)

Mitesh M. Khapra CS7015 (Deep Learning)

So, now it captures all the information from the previous state as well as the current input and we want to selectively read this. So now, what would you do to selectively read? Again the same situation that, you have a \tilde{s}_t the answer is already here. You have \tilde{s}_t and you do not want to pass it on as it is to s_t , this is \tilde{s}_t s_t is somewhere here, which you do not know how to get to, but you know that you do not want to pass on all the input that, you have read, you want to selectively pass it on. So, what will you do now? Again introduce a.

Student: Gate.

Gate and this gate will be called?

Student: Read gate.

Input gate or the read gate right ok.

(Refer Slide Time: 17:55)

The diagram illustrates the internal operations of a neural network cell. On the left, a matrix labeled 'selective write' is shown with columns h_{t-1} and x_t . This matrix is multiplied by a weight matrix W . The result is passed through a sigmoid function σ . On the right, a matrix labeled 'selective read' is shown with columns h_{t-1} and x_t . This matrix is multiplied by a weight matrix U . The result is passed through a sigmoid function σ . The final state s_t is the element-wise product of the two sigmoid outputs.

Selective Read

- \tilde{s}_t thus captures all the information from the previous state (h_{t-1}) and the current input x_t
- However, we may not want to use all this new information and only selectively **read** from it before constructing the new cell state s_t
- To do this we introduce another gate called the input gate

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$$

- and use $i_t \odot \tilde{s}_t$ as the selectively read state information

Mitesh M. Khapra CS7015 (Deep Learning) 21/43

So now, what can you give me an equation for the gate it is equal to sigma of that is good. Because, sigmoid is what we need it is going to be a fractional thing let me add the easy part W into.

Student: h_{t-1} .

h_{t-1} , that is telling you what has happened so far and U times x_t , you see the same equation, same form the parameters have changed. So, these we will call as $W_i U_i$ and b_i and they are depending on the input as well as the previous state, previous temporary state that we had computed ok. So, that is exactly, what your input gate is going to be and now this operation is the selectively reading operation, how many you are fine at this point? Ok and then this product is going to use to be it is will help us to read selectively from this temporary value that, we have constructed or the input that we have taken ok.

(Refer Slide Time: 18:51)

So far we have the following

Previous state:
 s_{t-1}

Output gate:
 $o_{t-1} = \sigma(W_o h_{t-2} + U_o x_{t-1} + b_o)$

Selectively Write:
 $h_{t-1} = o_{t-1} \odot s_{t-1}$

Current (temporary) state:
 $\tilde{s}_t = \sigma(W h_{t-1} + U x_t + b)$

Input gate:
 $i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$

Selectively Read:
 $i_t \odot \tilde{s}_t$

Mitesh M. Khapra CS7015 (Deep Learning) 22/43

So, far what do we have we have the following, we have the previous state, which was s_{t-1} , then we have an output gate, which was o_{t-1} using these 2, we have done selective write right, we have taken the previous state and the gate and done a selective write is that fine ok. We need to check, if the sigmoid should come here because, sigmoid is already there in the computation of s_{t-1} right it is not there. So, this already has 1 sigmoid right yeah. So then, again a sigmoid on that is it there we will figure it out, just check the equation right.

So, there may or may not be the sigmoid, the sigmoid already applied to s_{t-1} , but we can figure that out ok. So, this is the selective write portion then, you compute the current temporary state and just look at the similarity between these equations then, you have an input gate and using these 2, we have done a selective read ok. So, you have taken care of selective write and selective read, but you are still not reached s_t , I still do not have an arrow here, I still need to figure out, how to compute the s_t finally, ok. So, what is the operation which is remaining now? Selective.

Student: Forget.

Forget ok. So, what do you think should we forget? We want to find new s_t . So, let us see what we will forget right.

(Refer Slide Time: 20:07)

Selective Forget

- How do we combine s_{t-1} and \tilde{s}_t to get the new state
- Here is one simple (but effective) way of doing this:

$$s_t = s_{t-1} + i_t \odot \tilde{s}_t$$

• But we may not want to use the whole of s_{t-1} but forget some parts of it

Mitesh M. Khapra CS7015 (Deep Learning)

So, the question now is that you had this s_{t-1} and now you have a temporary state \tilde{s}_t which is here. How do we combine these 2 to get the new cell state? Ok. So, the simplest way would be that, you say that s_t is equal to whatever was there in s_{t-1} plus selectively reading from the current \tilde{s}_t input is that fine, this is the one way of doing it ok, but now what am I doing here, what is the problem here? I am reading, I am taking s_{t-1} as it is right. So, what should I do? I should forget some parts of s_{t-1} . So, what should I do for that introduce a what gate?

Student: Forget gate.

Forget gate right. So, we may not want to use s_{t-1} as it is, but we are want, to forget. So, there is at this point all of you should get some confusion, if you do not then, I would be worried, if you are getting some confusion good right. You should all get confused at this point, why are you confused? Because you already did selective write and now again you are doing a selective forget also right, but there is a difference because, the selective write was then used to compute, how to read the information right, but now once you have read the new information, you want to see how to assimilate it back with the old information that you had right. So, that is why you introduce a separate gate. So, think of it as this way that you are keeping these functions separate input, output and forget. So, they can separately learn things ok.

So, whatever you want to selective write let it be a separate function, these h_{t-1} is not going back to s_t right. Let us just by use so that, you can compute these temporary states. So, that is what is being passed to the next temporary state, let it only decide how much of this input should be read? Ok and then when you want to combine these 2, just use a separate gate and this exact idea, which is confusing all of you, why have a separate write gate and a separate forget gate? Led to something known as gated recurrent units, where they merged these to gates we will get back to that ok.

So, at this point it is fine. I am just telling you the original equations for LSTM and this was the motivation that they had. So, as I said there are at least 15 to 20 different variants of LSTM, which use different equations, they tie some of these weights. So, one thing could be that forget is the same as 1 minus remember right or output could be same as 1 minus input right, you could have tied these gates instead of learning separate parameters for that.

(Refer Slide Time: 22:57)

Selective Forget

- How do we combine s_{t-1} and \tilde{s}_t to get the new state
- Here is one simple (but effective) way of doing this:

$$s_t = s_{t-1} + i_t \odot \tilde{s}_t$$

- But we may not want to use the whole of s_{t-1} but forget some parts of it
- To do this we introduce the forget gate

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$$

$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t$$

Mitesh M. Khapra CS7015 (Deep Learning) 23/43

So, in the most parameterized form you have a separate parameter for all of these ok. So, we introduce the forget gate again, can you tell me the form for this forget gate f_t is equal to first term.

Student: W f.

What will be there in the second term x_t and the first term o_t . So, this is what it will look like. So, if you remember one of these equations, you will be able to write all of these, not that I am going to ask you to write them in quiz or something, but why take a chance. So, and then once you have completed the forget gate, instead of this equation, can you tell me what is the equation and you are going to use? What is the first term going to be it is s_{t-1} here? What is it going to be now?

Student: f_t into.

f_t into?

Student: s_{t-1} .

s_{t-1} that fine ok.

(Refer Slide Time: 24:47)

- We now have the full set of equations for LSTMs
- The green box together with the selective write operations following it, show all the computations which happen at timestep t

Gates:
 $o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o)$
 $i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$
 $f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$

States:
 $\tilde{s}_t = \sigma(W h_{t-1} + U x_t + b)$
 $s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t$
 $h_t = o_t \odot \sigma(s_t) \text{ and } rnn_{out} = h_t$

Handwritten notes: h_t, s_t, c_t

Mitesh M. Khapra CS7015 (Deep Learning) 24/43

So now, we have the full set of equations for LSTM, we have certain gates and certain states, what are the gates? Output gate.

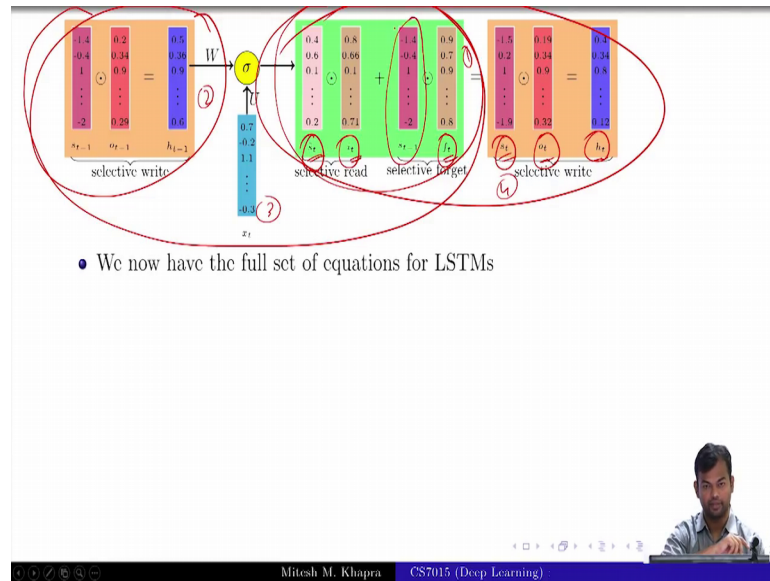
Student: Input gate.

Input gate.

Student: Forget gate.

Forget gate, why do you guys has this momentary amnesia like suddenly you forget everything ok. So, output gate, input gate and forget gate all of these are the same form with different parameters ok. What about the states, which are the states that we have completed? One was s_t the other was h_t and the third one was s_{t-1} ok. s_{t-1} from s_{t-1} , we get s_t and from s_t , we compute h_t ok.

(Refer Slide Time: 24:37)



So in the diagram, that you see here at the top. Tell me which are the computations, which are happening at one time step at time step t , which are the computations, which are the computations, which are happening? Is it I will give you the options right, is it this or is it this let us call this 1, let us call this 2 or this 3 or this 4. Which are the computations happening at one time step? And you see the order also here this should be straight forward right why?

Student: (Refer Time: 25:16).

How many of you say 4? That is the one right because, you start with selective reading right and you can just go by this right these are all indexed by t right is that fine ok. So, these are the computations, which happen at time step t and these are exactly the computations which were written right. So, we have the 3 gates, which you need to compute at every time step and you have the 3 states, which you need to compute at every time step is that fine and this s_{t-1} is not being computed. It is just taken from the previous time step is that fine. So, you have these 6 computations, which

happened at every time step and the output final output of an LSTM. So, when you use tensor flow or something the output of an LSTM, would give you 2 things. It will give you h_t comma s_t because, these both the states that are being computed one is the running state and another one is the current state, which is being computed ok.

And I choose the notation S because, that is what we have been using for RNNs, but in LSTM in all the literature instead of S , you will find it to be ct because it is called the cell state. So, that is why s_t ok. So all these equations wherever, you see an s when you are reading some standard blogs or things like that you will see C instead of S . So, you just do this mapping in your head ok.

(Refer Slide Time: 26:41)

Note

- LSTM has many variants which include different number of gates and also different arrangement of gates
- The one which we just saw is one of the most popular variants of LSTM
- Another equally popular variant of LSTM is Gated Recurrent Unit which we will see next

Mitesh M. Khapra CS7015 (Deep Learning)

So, LSTM actually has many variants with include different number of gates and also different arrangement of the gates. So, as I was saying that you could say that input is 1 minus output or input is 1 minus forget or things like that and also why this particular parametric form right why not make W_0 into s_t minus 1 instead of h_t minus 1 and so on. So, the all points, of things that you could do or all of these are valid these are all valid variants of LSTMs.

So, there is this paper called LSTM, a search space odyssey. So, you can go and look at I think we link it in the in the reading material right ah. So, you can see that there are actually many many variants of LSTMs, but this is the most standard and default variant, which you will find in most platforms on tensor flow or (Refer Time: 27:26) form.

(Refer Slide Time: 27:33)

The full set of equations for GRUs

Gates:

$$o_t = \sigma(W_o s_{t-1} + U_o x_t + b_o)$$

$$i_t = \sigma(W_i s_{t-1} + U_i x_t + b_i)$$

States:

$$\tilde{s}_t = \sigma(W(s_{t-1} \odot s_{t-1}) + U x_t + b)$$

$$s_t = (1 - i_t) \odot s_{t-1} + i_t \odot \tilde{s}_t$$

- No explicit forget gate (the forget gate and input gates are tied)
- The gates depend directly on s_{t-1} and not the intermediate h_{t-1} as in the case of LSTMs

NPTEL Mitesh M. Khapra CS7015 (Deep Learning) 26/43

And there is another very popular variant of LSTMs, which is called gated recurrent units. So, we will just see gated recurrent unit. So, I will just give you the full set of equations for GRUS. So, you have gates, but unlike LSTMs, you have only 2 gates, you have an output gate and you have an input gate, you do not have the forget gate ok.

So, what am I going to do for the forget gate? So, this is what I am going to do, you see the last equation. So, instead of forget gate I am just saying that this is what you are going to selectively input from the current temporary state. So, the rest of you rest of it you take from the previous state right. So, I have just tied the input gate and the forget gate any other changes do you see in this. So, earlier we had s t minus 1, everywhere right now we have s t minus 1 itself is that fine ok. So, the basic idea these equations are many many and you could think of your own equations, you could say that I will not really use this input information at all or I will choose to use it differently or what not right there are several things that you could do. At a very abstract level this is what you need to, what is this?

Student: (Refer Time: 28:38).

So, these parameters could then make a difference right, they could adjust it accordingly and so 1 right. So, that is what I was coming. So, the there are various ways of realizing this right at the abstract level, you need to understand that the original problem was trying to store all the information from time step 1 to time step T capital T right, which is

not feasible, because of this finite size that you have. So, along the way we built this intuition that it should be good to have these operations, which allow you to selectively read write and forget right. How do you mathematically realize these operations? There are various choices for doing that and we saw a few choices for doing that right, there are many others you could have done. But this is largely, what whenever you say that I have used an LSTM, most likely you are using the set of equations which I saw, which we saw on the previous slide and whenever you are using a GRU, these are the set of equations that you will be using ok.

And again remember this, that there is no explicit supervision here, it is just that we have a better modelling choice we are just introduce more parameters. So, that if required these parameters could be adjusted to do a selectively read, write and forget right. So, it is often, it is often valuable, if you are doing some task with RNNs or LSTMs. You should visualize these gates right, you should see that at time step t , if you thought that it should have forgotten everything that it has learnt. So far, because suppose you had this the movie was long, but I really loved it because, the direction was superb and so on.

Now this word, but actually changes everything right because, it whatever was written before it does not matter anymore right. So, is it really learning those kind of gates, where everything before, but was forgotten right. So, it would be helpful to visualize these output gates and see what kind of values, they are learning. What kind of things they are remembering forgetting and selectively reading and so on right. So, as I said I will just again summarize the key thing here, is the intuition and then the realization in the form of equations, there are multiple choices, we have seen a few of those right that is what I will end with. And, in particular in GRUS, there is no explicit forget gate and instead of h_{t-1} you use s_{t-1} everywhere.